2.登陆页面制作

1.引入Element –ui

官网:https://element.eleme.cn/#/zh–CN/component/installation

安装

npm 安装

推荐使用 npm 的方式安装，它能更好地和 webpack 打包工具配合使用。

```
npm i element-ui -S
```

在终端运行命令npm i element–ui –S

在main.js加入以下命令

XML 复制代码

```
import ElementUI from 'element-ui';
import 'element-ui/lib/theme-chalk/index.css';

Vue.use(ElementUI);
```

完整引入

在 main.js 中写入以下内容：

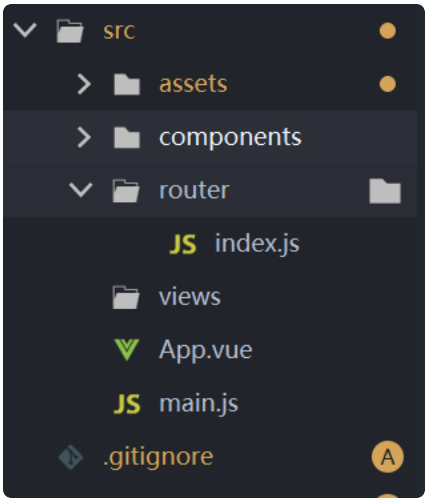
```
import Vue from 'vue';
import ElementUI from 'element-ui';
import 'element-ui/lib/theme-chalk/index.css';
import App from './App.vue';

Vue.use(ElementUI);

new Vue({
  el: '#app',
  render: h => h(App)
});
```

2.目录整理

清理目录,将不需要的文件删除(About Home helloworld)



清理App.vue

```vue
<template>
  <div id="app">

    <router-view/>
  </div>
</template>
```

3.页面创建

在view目录下创建login.vue

Vue | 复制代码

```vue
<template>
  <div>
    <el-form :rules="rules" :model="loginForm"  class="loginStyly">
      <!-- rules校验规则   model表单数据对象,设置登陆数据 -->
      <h3 class="loginTitle">请登录</h3>
      <el-form-item prop="username">
        <el-input type="text" v-model="loginForm.username"   auto-complete="off" placeholder="请输入用户名"></el-input>
      </el-form-item>
      <!-- prop一般用于父组件向子组件进行单向通信（父向子传值），传递的可以是一个对象，也可以是一个数组。
      prop是数组的话会有默认的数据类型，是对象的话可以设置数据类型，是否必传等 -->
      <el-form-item prop="password">
        <el-input type="text" v-model="loginForm.password"   auto-complete="off" placeholder="请输入密码"></el-input>
      </el-form-item>
      <!-- v-model="loginForm.password"  绑定  loginForm的值   auto-complete="off   是否自动填充-->

      <el-checkbox v-model="check" class="loginRemember">记住我</el-checkbox>
      <el-button type="primary" style="width: 100%;">登陆</el-button>
      <!--
      danger红色
      info 灰色
      primary 蓝色
      success 绿色
      text 透明
      warning 黄色
      -->

    </el-form>


  </div>
</template>

<script>
  export default{
    name:"login",
    data(){
      return{

        loginForm:{
          username:'',
```

```
            password:''
          },
          check:true,
          rules:{
            username:[{required:true,message:"请输入用户名",trigger:'blur'}],
            password:[{required:true,message:"请输入密码",trigger:'blur'}]
          }
        }

      }
    }
  </script>

  <style>
    .loginStyly{
      border-radius: 15px;/*边框*/
      background-clip: padding-box;/*确定背景裁剪区域*/
      margin: 180px auto;
      width: 350px;
      padding: 35px 35px 15px 35px;
      background-color: aliceblue;
      border: 2px solid #fff;
      box-shadow: 0 0 25px #aaa;
    }
    .loginTitle{
      margin: 15px auto 20px auto;
      text-align: center;
      color: aqua;
    }
    .loginRemember{
      text-align: left;
      margin: 0px 0px 15px 0px;
    }
  </style>
```

更改index.js

```
import Vue from 'vue'
import VueRouter from 'vue-router'
import login from '../views/login.vue'

Vue.use(VueRouter)

const routes = [
  {
    path: '/',
    name: 'login',
    component: login
  }
]

const router = new VueRouter({
  routes
})

export default router
```

4.知识点

1.v-model:和:model

v-model 是 v-model:value 的缩写，通常用于表单上的双向数据绑定（表单接受值 value，故v-model默认收集的就是 value ，所以缩写直接省略 value），可以实现子组件到父组件的双向数据动态绑定。数据不仅能从data流向页面，还可以从页面流向data。

:model:

:model 是 v-bind:model 的缩写，可以实现将父组件的值传递给子组件，但是子组件不能传给父组件，无法双向绑定。

2.Background-clip

1.作用：CSS3中的Background-clip属性，其主要是用来确定背景的裁剪区域，换句话说，就是如何控制元素背景显示区域

2.语法：background-clip： border-box ‖ padding-box ‖ content-box

3.取值说明：

1.border-box:此值为默认值，背景从border区域向外裁剪，也就是超出部分将被裁剪掉

2.padding-box：背景从padding区域向外裁剪，超过padding区域的背景将被裁剪掉

3.context-box：背景从content区域向外裁剪，超过context区域的背景将被裁剪掉

3.margin

margin的用法：

1：控制的是盒子与盒子之间的位置关系

2：margin长在盒子外围的，不会对盒子本身的大小造成影响。

3：给单一一个方向添加margin值：margin-left/right/top/bottom:

4:margin设置方法：

margin:10px 四周

margin:10px 20px 上下 左右

margin:10px 20px 30px 上 左右 下

margin:10px 20px 30px 40px 上右下左

5:margin支持负值！！

6:让子元素在父元素里面左右居中：margin:0 auto;

7:margin常见的bug:

a:当父元素和子元素都没有浮动的情况下：给第一个子元素添加margin-top:会错误的把margin值加在父元素上面

b：相邻两个元素上下margin会重叠，按照较大的值设置。

4.padding

1:padding是添加在父元素（盒子）上的

2:padding 调整子元素在父元素里面的位置关系

3:padding会把盒子撑大。

4：想让盒子保持原有的大小：在宽高的基础上减掉padding值。

5:给单一一个方向添加padding值： padding-top/bottom/left/right:

6:padding设置方法：`在这里插入代码片`

padding:10px 四周

padding:10px 20px　上下 左右

padding:10px 20px 30px　上　左右　下

padding:10px 20px 30px 40px　上右下左

7：padding不会对背景图造成影响

8：padding的值不能为负值！！！

5.background–color

背景颜色

6.border

首先border共有四种 分别是dotted solid double dashed

单实线　border: 3px solid red;

效果图

双实线　border: 3px double red;

效果图：

点虚线　border: 3px dotted red;

效果图:

短线虚线　border: 3px dashed red;

效果图:

7.box-shadow

Vue　复制代码

```
box-shadow: none ;

box-shadow: inset  x-offset  y-offset  blur-radius  spread-radius  color;
```

阴影类型：此参数可选，默认的投影方式是 外阴影；如果取其唯一值“inset”,就是将外阴影变成内阴影

X-offset：是指 阴影水平偏移量 ,其值可正可负，正值，则阴影在对象的右边，负值，阴影在对象的左边

Y-offset：是指 阴影的垂直偏移量 ，其值也可以是正负值，正值，阴影在对象的底部，负值时，阴影在对象的顶部

阴影模糊半径：此参数是可选，只能为正值，如果其值为0时，表示阴影不具有模糊效果，值越大阴影的边缘就越模糊

阴影扩展半径：此参数可选，其值可为正负值，正值，则整个阴影都延展扩大，反之，则缩小

阴影颜色：此参数可选，不设定任何颜色时，浏览器会取默认色，但各浏览器默认色不一样，特别是在webkit内核下的safari和chrome浏览器将无色，也就是透明，建议不要省略此参数

注：多层阴影，最内层优先级最高，之后依次降低。使用逗号“，”隔开。

举例 1

不设置X轴与Y轴，设置值阴影模糊半径为15px, 它会在本身发生作用 半径范围，颜色

Vue 复制代码

```
box-shadow: 0 0 15px #f00;
```

举例 2

X轴与Y轴设为正值（正值 X轴向右 Y轴向下）

Vue 复制代码

```
box-shadow:4px 4px 15px #f00;
```

举例 3

box-shadow：inset 即box-shadow内部阴影，与上面写法相同 唯一不同的是添加了一个inset

Vue 复制代码

```
box-shadow:0 0 15px #f00 inset;
```

举例 4

设置正方形的四边颜色都不一样，但是阴影模糊半径都为10px

```Vue
box-shadow:-10px 0px 10px red,   /*左边阴影*/

            0px -10px 10px black,  /*上边阴影*/

            10px 0px 10px green,  /*右边阴影*/

            0px 10px 10px blue;" /*下边阴影*/ >
```

8.text-align

left,right,center,这三个取值，想当直接，就是居左，居右，据中。其中，在不设置text-align属性时，浏览器默认是居左的。

justify,两端自适应对其。这个时候，会根据父元素的宽度，进行自适应的两端对其。
inherit，规定应该从父元素继承text-align属性的值。
虽然说，在不设置text-align的情况下，浏览器是默认居左的，但是其前提是，它的祖辈元素，也没有设置text-align属性，如果祖辈设置了，那么目标元素的text-align是继承它父元素的text-align属性的。

1：首先说明，大家都知道， 元素的表现形式主要可以分为：块级元素，和行内元素。 text-align属性的设置，对于块级元素是无效的，但是对于行内元素是可以起作用的。所以，一个元素的显示对齐方式，就可以由两个因素决定，父元素的text-align属性，和自身的display属性。

注：text-align属性，只对子元素起作用，并且可以被子元素继承。display属性，只对本身起作用，并且不可被子元素继承。